241-TP-001-001

Systems Performance Models for the ECS Project

Technical Paper

Technical Paper--Not intended for formal review or government approval.

June 1996

Prepared Under Contract NAS5-60000

RESPONSIBLE ENGINEER

E. H. Brackett /s/ 7/1/96

E.H. Brackett, System Performance Modeling Date
EOSDIS Core System Project

SUBMITTED BY

R. E. Clinard /s/ 7/1/96

Robert Clinard, System Management Office Manager Date
EOSDIS Core System Project

Hughes Information Technology Systems
Upper Marlboro, Maryland

This page intentionally left blank.

Abstract

The Systems Performance models for the ECS project assist in the design of the ECS system by providing static analysis, a steady state queuing model, and dynamic simulation of the major components of the system architecture. The models allow exploration of alternative designs of capacity and end-to-end response times for processors, communications, and levels of storage with varying hardware characteristics and workloads.

Keywords: Modeling, performance, capacity, response times, dynamic, system, analytic

This page intentionally left blank.

Contents

1. Introduction

2. Static Models

3. Dynamic Model General Description

4. Dynamic Model Function

5. End-to-End Model

Figures

This page intentionally left blank.

1. Introduction

1.1 Purpose

The purpose of this paper is to provide a description of the ECS System Performance models.

1.2 Scope

The Systems Performance Models are intended to represent only the highest levels of the ECS. Models of individual subsystems and other more detailed models may be developed as the need arises. Those models are not documented here.

1.3 Organization

This paper is organized as follows:

- Section 1 contains the introduction to this paper.
- Section 2 contains a summary of the Static Models.
- Section 3 contains a general overview of the Systems Performance Dynamic Model.
- Section 4 contains a detailed description of the Dynamic Model.
- Section 5 contains a description of the End-To-End Model.
- Appendices A, B, and C contain PDR and CDR Baseline data.
- Acronyms and Abbreviations.

Questions regarding technical information contained within this Paper should be addressed to the following ECS contact:

- Hal Brackett, SI&P Performance Modeling, (301) 925-0511, hbracket@eos.hitc.com

Questions concerning distribution or control of this document should be addressed to:

Data Management Office
The ECS Project Office
Hughes Information Technology Systems
1616 McCormick Drive
Upper Marlboro, MD 20774-5372

This page intentionally left blank.

2. Static Models

2.1 Static Models Overview

The major static model is for first-time push processing and is implemented as a spreadsheet. The model is used to provide insight into the average and "busy day" magnitudes of the processing CPU and I/O loads on the SDPS. This model is the first one executed for new push data and is the first step in sizing the SDPS.

The ECS Science Office maintains a separate static model of the inter-DAAC traffic.

2.2 Push Model Input

All input data is from the Technical Baseline for the ECS Project: the operating hours by site; and, the AHWGP process description file for each site, instrument, and time period (epoch) target that characterizes the push load on the system in terms of I/O volumes, PGE execution times and frequency of invocation.

2.3 Push Model Calculations

The process description file is sorted in order by epoch and instrument. The average number of MFLOPS is calculated for each PGE by:

$$t_{cpu} = MFPOs_{process} / MFLOPS_{derated}$$

$$MFLOPS_{derated} = MFLOPS_{vendor} / factor_{derating}$$

The average I/O bandwidth required for staging and destaging the data for each PGE is calculated. Results are accumulated for each instrument by site and by epoch.

The same values are recalculated for the "busy day". A busy day is when all PGEs with frequency of execution of less than once per day, are caused to execute on the same day as the daily PGEs. This has the effect of simulating a day when all products need to be produced on the same day.

2.4 Push Model Output

The results provide analyses of average and busy day and provide summaries for each instrument by epoch and by site for: the number of PGE invocations per day, the total MFLOPS required; and, the I/O bandwidth requirements (MB/second) for the local disk to processing, the host-attached backplane, and combinations of staging and destaging.

This page intentionally left blank.

3. Dynamic Model General Description

The System Performance Model (referred to as the "model" throughout Sections 3, and 4) was developed as a system simulation using the Block Oriented Network Simulation (BONeS)®[1] tool. It is designed as a dynamic model to support capacity planning, requirements analysis, design, and development of the ECS. The model includes the performance of external elements and is updated in parallel with the system development, simulating the as-developed system to allow performance checking of the completed system and evaluation of any changes proposed as modifications to the completed system. It is sufficiently detailed to permit it to be used to select and validate processor hardware and software architectures. It also can be used to simulate data flows from instruments to investigators, user interactions with the ECS or with individual instruments, and the processing workload resulting from these activities.

3.1 Dynamic Model Overview

Figure 3-1 shows the primary interfaces for the Dynamic Model. The major input interfaces are with the Ad Hoc Working Group for Production (AHWGP), the User Model, and with the ECS Release design teams. Technology characteristics and projections are also input.

Through the AHWGP, the Instrument Teams (ITs) have provided scenarios for the generation of standard products in the form of process descriptions, timelines of process activations, and file descriptions of the size and types of files. Results from the dynamic model and validation of the input data are returned to the AHWGP and ITs. The interaction of the performance modeling team with the AHWGP and ITs takes two forms: through the Algorithm Support Teams (ASTs) in the ECS Science Office and one-on-one with individual scientists within the ITs. The interactions with the ASTs allow information from the science community to be baselined by the ECS Chief Engineer. The one-on-one interactions allow exchange of information on improvements that may be made in the execution of the science algorithms and how to realize such optimizations.

The User Model provides a characterization of the science user community in terms of request profiles.

From the release design teams, modeling receives design and operational concepts for each subsystem. Release design teams also provide performance data on current and proposed hardware configurations and software technology. From this information, modeling creates and instruments (or modifies existing) simulations. The results of the simulation runs and subsequent analyses are provided as technical reports and iterated with the design teams for alternatives analysis.

---

[1] ® BONeS is a registered trademark of the Alta Group of Cadence Design Systems, Inc.

The technical reports, along with status reports, are provided to ESDIS and IV&V. Results of the IV&V of the model are returned via ESDIS.

The modeling results and analyses are provided to the Cost Model.

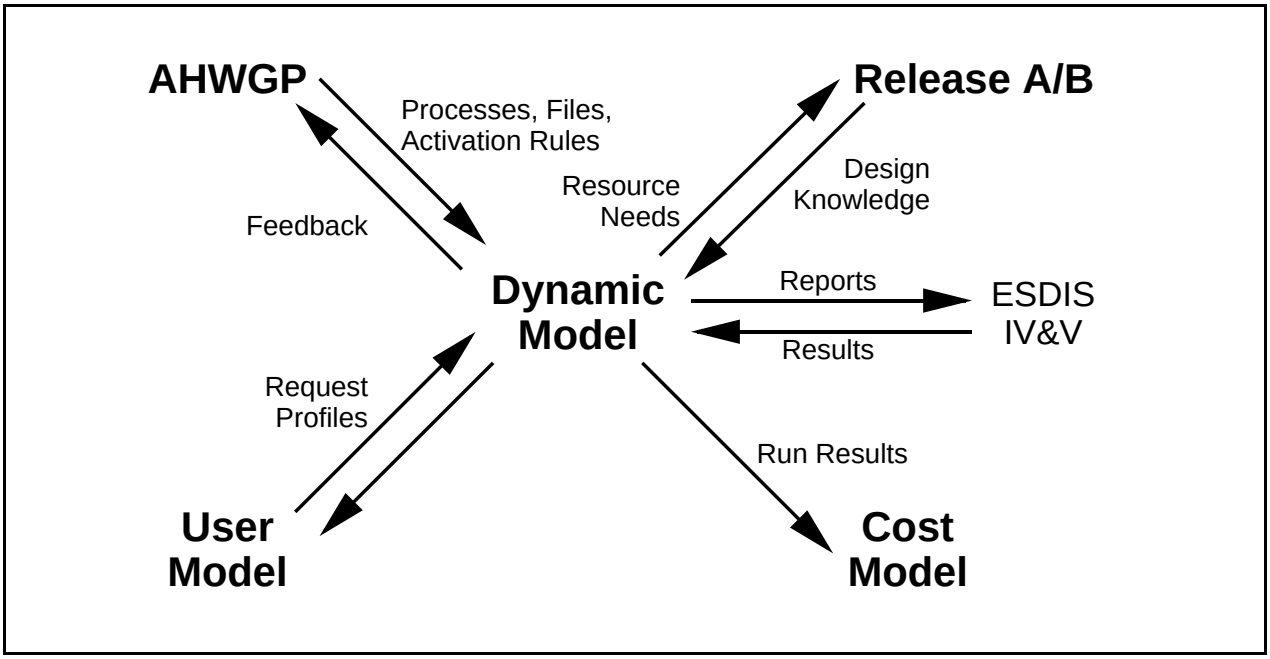

Figure 3-1. Performance Model Context

Figure 3-2 is a high level overview of the performance model in terms of the major interfaces among the subsystems. The model uses multiple copies of this diagram - a copy for each DAAC. Each subsystem is represented in the model by a software module. The model is driven by the Push and Pull Generators, the Reprocessing Generator and the V0 Data Migration Generator .

The Push Generator provides the input interface for the platform-instrument data such as the AHWGP data. The Push Generator output is input to the Ingest module.

The Pull Generator is the input interface for the User Model data. The Pull Generator user processing requests are input to the Event Driven Scheduler while retrieve requests are input to the Data Handler. This module interface is detailed in Section 4.4.

The Reprocessing Generator has several paradigms to duplicate the original data stream as an input to the Data Driven Scheduler.

The V0 Data Migration Generator provides the input interface for the data being migrated from the Version 0 system. The migration requests are input to the Data Driven Scheduler.

The Ingest module provides the Level 0 data directly to Processing for archive and passes all other data to the Data Handler. The Data Handler module provides staging of data between the archive and all other modules to obtain the data needed for input by the process and to return data from the process to the archive.

The Processing module simulates the production of all products by accounting for the execution time required to produce the product.

The Event Driven Scheduler determines what should be scheduled and when it should be scheduled, from the requests it receives, the data availability, and the process activation rules.

The Distribution module is responsible for DAAC output to local media and by electronic media to the other DAACs and the SCFs.

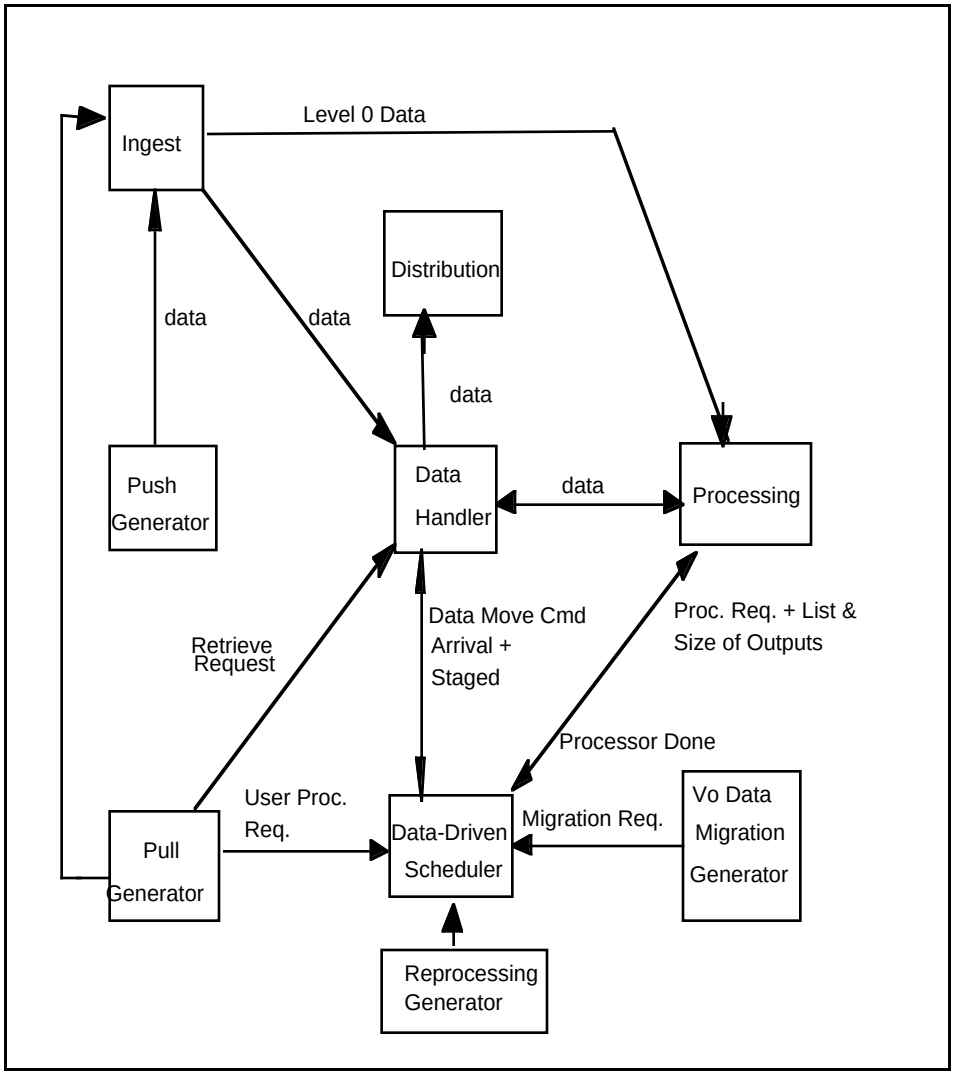

Figure 3-2. Performance Model Overview

3.2 Parameters

At model Initialization, the preset general parameters for the model are divided into two categories: Processing and Networks.

The parameters for processing are defined per processing pool (or string). Each pool has a settable number of processors, processor power expressed in MFLOPS or MIPS, and a settable total disk pool size expressed in megabytes. Also determined is the total number of I/O channels and the bandwidth of I/O channels expressed in MB/second.

Networks are modeled both within each DAAC (LAN) and between the DAACs (WAN). Each has a flag indicating if the network is shared or is exclusive to the DAAC. Network efficiency, expressed as a fraction, is defined, as is the bandwidth of the Network, expressed in MB/second.

3.3 Input

The workload input to the model is specified in the Technical Baseline for each formal review as detailed in the Appendices. Inputs to the model are derived from three primary sources:

- User Model/AHWGC data;
- AHWGP data;
- SPSO data.

3.3.1 User Model /Pull Generator

Parameters have been established for the Pull Generator on a "per epoch" basis. For each epoch, user model values are provided for four variables (Figure 3-3). First, service interval time vs. time of day. Second, fraction of transactions vs. DAAC. Third, fraction of transactions vs. service. Last, service specific parameters.

The model operates on a predetermined access pattern of users throughout the workweek or weekend day, with an established mean time between transactions vs. time table, for all transactions. The variables are statistically decoupled into the number in each user class for EOS and other science users in two formats: the percentage of transactions directed to each DAAC per user class, and the percentage of transactions for a particular service per DAAC per user class. There is a further decoupling for the distribution of unique characteristics of each service request per service. This decoupling process is illustrated in Figure 3-3 for a typical epoch.

3.3.2 AHWGP Input Data

The AHWGP has established parameters as well as provided a suite of input data. The parameters are divided into per file and per process categories. On a per file basis, they have defined the file ID/name format; the file size; the file disposition (archive, interim, permanent, temporary, QA, SCF, or user); and the archive site. On the per process basis, parameters have been established for process ID/name format; process execution site; the applicable epoch(s); the list of input file ID's, including the number and amount read; the list of output file ID's, including the number and amount written; and the processing time, measured either by wall clock or MFLOPS/MIPS as MFLO. An epoch is one quarter of a year; and is identified by lower case letters with epoch a equal to 1Q97, epoch k equal to 3Q99, through epoch x equal to 4Q02.

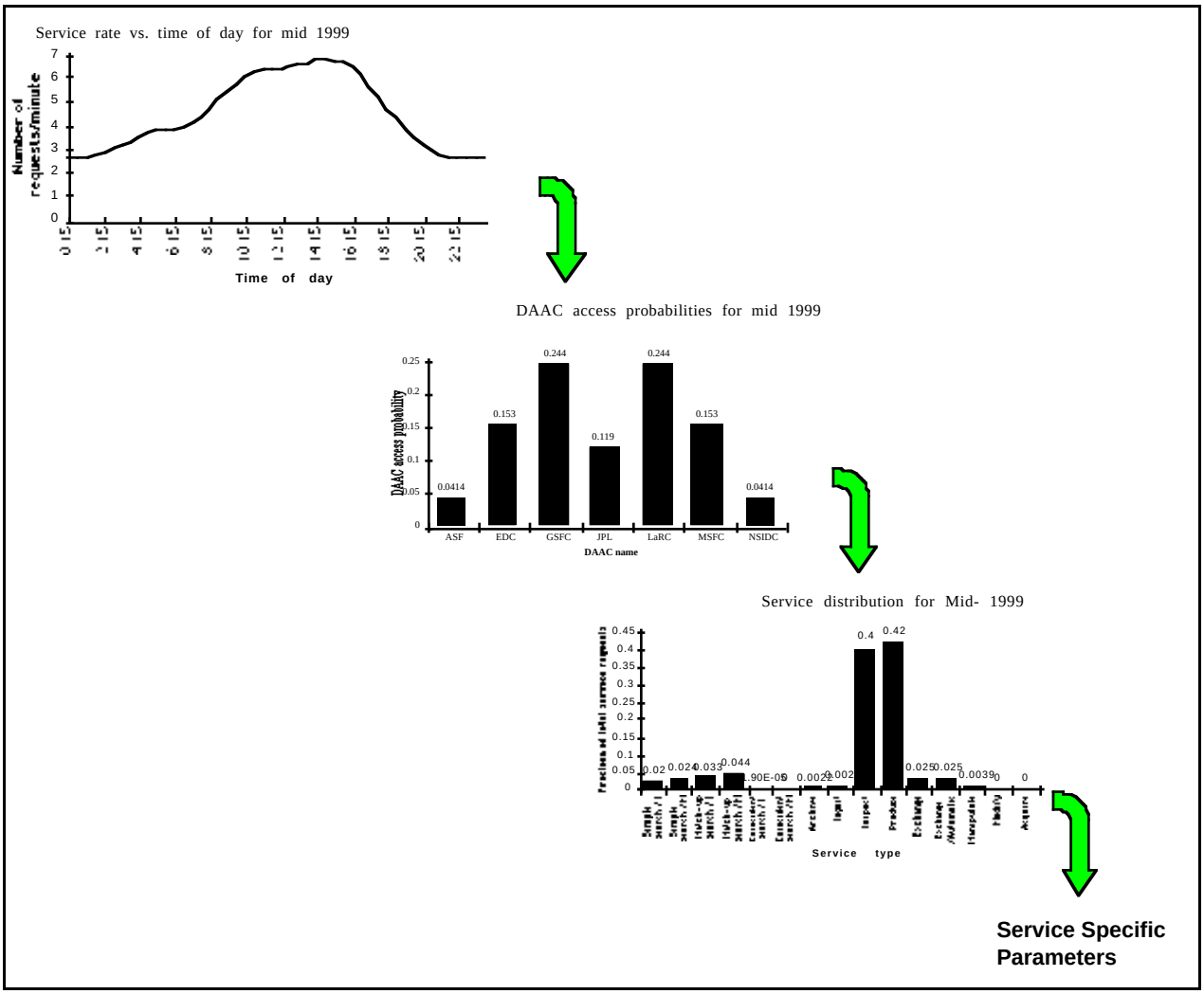

Figure 3-3. User Model Variables are Statistically Decoupled

3.3.3 V0 Input Data

The V0 data migrated is characterized only by volume.

3.3.4 Reprocessing Generator

The reprocessing load is simulated by a set of macros, each of which generate a specific loading. Current macros include: doubling the frequency and halving the coverage; and, duplicating the primary workload at a lower priority. Both assume that the standard product processing sequence is repeated from the beginning.

3.4 Module Input Parameters

The parameters used by the simulation modules are summarized in Figure 3-4 and discussed in the following sections.

	Disk	Processors	I/O Channels	Network	Robots	Read/Write Heads
Ingest	Y		Y	Y	Y	Y
Data Handler	Y		Y	Y	Y	Y
Processing	Y	Y	Y	Y		
Distribution	Y	Y	Y	Y		Y

Figure 3-4. Components in Simulation Modules

3.4.1 Ingest Module

All data for the Ingest Module is arranged by DAAC. The parameters, per DAAC, for the module include the total archive disk pool size (in MB), the total number of I/O channels, the bandwidth of I/O channels (in MB/sec), the bandwidth of the network (MB/sec), the number of robots, the maximum robot movement time (in seconds), the number of read/write heads, the maximum tape seek time (in seconds), the number of I/O channels for each archive device, the bandwidth of read/write heads (in MB/sec), and the maximum tape rewind time (in seconds). In addition, for all files received via the physical media for each media type, the number of read/write heads and the maximum delay time to mount/dismount (in seconds) is defined.

3.4.2 Data Handler Module

Information for the Data Handler is supplied on a per DAAC basis. The parameters established for this module include the total archive disk pool size (in MB), the total number of I/O channels, the bandwidth of I/O channels (in MB/sec),the bandwidth of the network (MB/sec), the capacity of the processors (in MFLOPS). the number of robots, the maximum robot movement time (in seconds), the number of read/write heads, the maximum tape seek time (in seconds), the number of I/O channels for each archive device, the bandwidth of read/write heads (in MB/sec), and the maximum tape rewind time (in seconds). Additionally, parameters are set for the number of transactions into each archive per data server and the storage utilization per DAAC.

3.4.3 Processing Module

The parameters for the Processing Module are also defined by DAAC. Included in the parameter list is the total distribution disk pool size (in MB), the capacity of the processors (in MFLOPS), the total number of I/O channels, the bandwidth of I/O channels (in MB/sec), and the bandwidth of the network (MB/sec).

3.4.4 Distribution Module

The parameters for the Distribution Module are also defined by DAAC. Included in the parameter list is the total distribution disk pool size (in MB), the capacity of the processors (in MFLOPS), the total number of I/O channels, the bandwidth of I/O channels (in MB/sec), and the bandwidth of the network (MB/sec). File disposition for user, QA/SCF, and all others, is defined as a percentage of files to each media type, including the network. For files distributed via an electronic network, the percentage of files pushed, the percentage of files pulled, and the maximum file pick-up wait time for files pulled (in hours) are defined. Files distributed via physical media for each media type are broken out into the number of read/write heads, the maximum delay time to mount/dismount (in minutes), and the bandwidth of the read/write mechanism (in MB/sec).

3.5 Outputs

The outputs of the Dynamic Model simulate those of the project. Output data is collected by the simulation with probes. Probes can be inserted for specific events to collect data about the event at the measurement points shown in Figure 3-5. The data collected by the probe(s) is summarized at the end of the simulation. For example, the time that a user request entered the system and the time the result was returned is collected by the probes and summarized to obtain the response time. A series of user requests can be summarized to obtain an average and maximum response times.

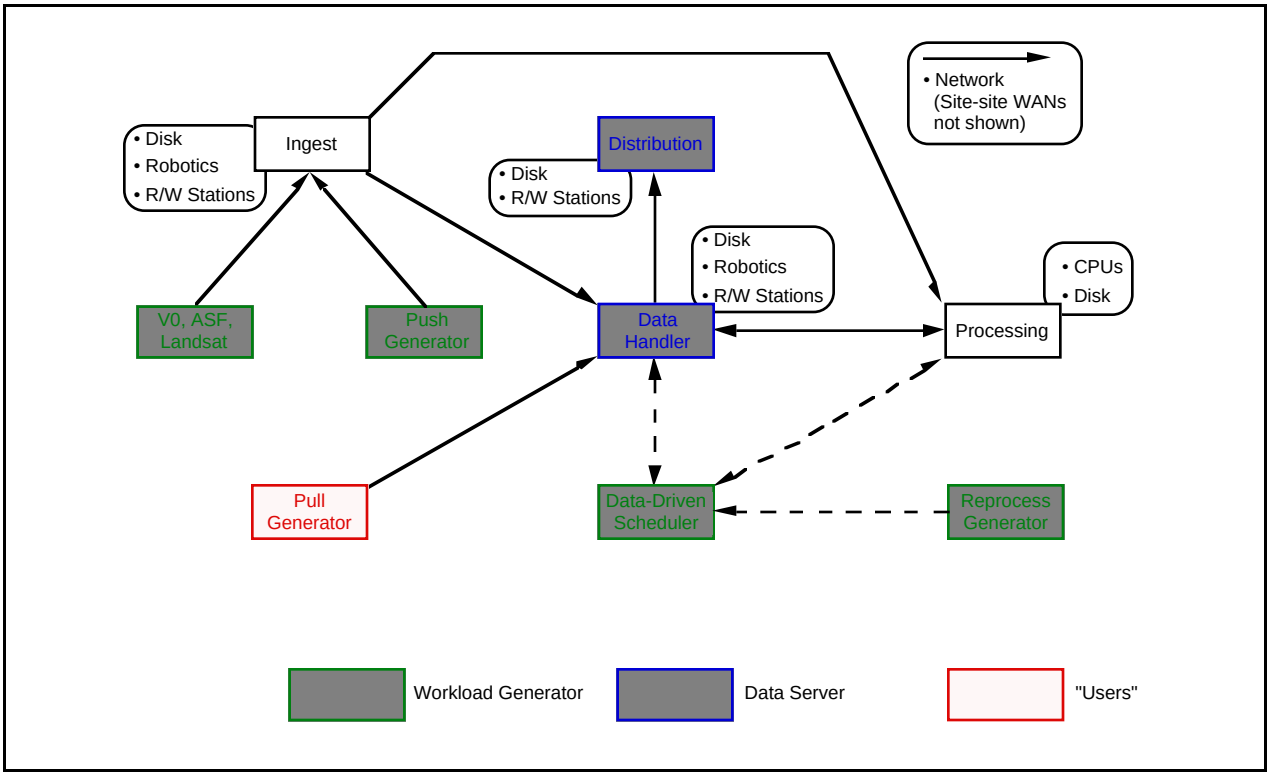

Figure 3-5. Measurement Points

Figure 3-6 shows some of the values measurable by the model for the physical components of a system configuration.

Figure 3-7 provides the name and units of the parameters associated with the physical components of a system configuration.

	Disk	Processors	I/O Channels	Network	Robots	Read/Write Heads
Amount Used	Y			Y		
Utilization Fraction	Y	Y	Y	Y	Y	Y
Number Used		Y	Y		Y	Y
Queue Occupancy	Y	Y	Y	Y	Y	Y

Figure 3-6. Metrics on Components

	Disk	Processors	I/O Channels	Network	Robots	Read/Write Heads
Size (MB)	Y					
Number (Count)		Y	Y		Y	Y
Power (MFLOPS)		Y				
Latency (Seconds)					Y	Y
Throughput (MB/S)			Y	Y		

Figure 3-7. Parameters Associated with Components

3.6 Assumptions

3.6.1 Primary Assumptions

A more detailed description of the modeling assumptions is available from the ECS Modeling Workshop #1 documentation, 731-001-001. There, assumptions are presented by the categories - system, input, distribution, archive, processing, reprocessing, and modeling - with the rationale and impact for each assumption.

The System Performance Model was built using five primary assumptions.

Assumption 1

EDOS will pass the instrument data to ECS in time-delimited sets of packets that cover the same amount of time as the Level 1A granules.

The current operations concept calls for EDOS to be able to package instrument data into time-delimited groups of packets. The grouping of the packets in this fashion allows for a smooth transition of data into the Level 1A algorithms, since the algorithms work with their input data in groups of this size. If the data comes in a different grouping, ECS will have to be able to perform the EDOS function of grouping the packets into time-delimited sets of packets. This would increase the processing and disk requirements within ECS.

Assumption 2

For each Instrument, each Level 0 data is used by (that is, input to) only one process.

To date, no example of Level 0 data feeding more than one Level 1A process has been found. Should the Level 0 data be required by more than one process, the amount of "waiting" storage would have to be increased. It might also be necessary to increase the capacity of the rolling storage hardware to account for any retrieve requests.

Assumption 3

Unless explicitly told otherwise by the algorithm development team, a Single Process Generation Execution (PGE) runs on only a single processor within a computer.

The heritage PGE being used by the instrument teams is designed for single processors. Those teams that do not have heritage code are developing their PGE for single processors. Should this assumption change, the number of processors needed would not change, but the amount of local storage might. The actual amount would have to be determined be analyzing specific products.

Assumption 4

For both the Ingest and the Data Server, the tape operations at the archive are not optimized. That is, no attempt is made to organize data sets onto individual tapes.

Depending on user and production access patterns, tape organization may lower the number of robots and read/write stations required. Access times may also be lowered. Some tape organization schemes may increase the amount of staging disk required.

Assumption 5

Both the subsetting and user-specified processing operate on data that is available from the data in the archive. The ECS is designed to facilitate EOS research. If a user is submitting data which neither reads nor writes EOS data and wants his own algorithm run on the data, then ECS is being used only as a processing resource. The impact of such an occurrence is largely unknown, since it would depend on what data is submitted for processing and the nature of the algorithms. Such impact could vary from slight to large.

3.6.2 Implementation Assumptions

1. The pull model is stochastic.
2. A PGE (Product Generation Execution) is the smallest independently schedulable processing unit.
3. The push model is completely deterministic in the sense that processing times, files sizes, and interactivation intervals are all AHWGP input.
4. The push model is data driven, i.e., PGEs are only scheduled when all necessary input files are present. A PGE can only execute when the necessary files are local to its CPU.
5. Each machine is assumed to have multiple CPUs.
6. Each CPU is dedicated to a single PGE.
7. Individual CPUs are not multiprogrammed.
8. There is no parallel processing: a PGE runs in a single CPU.
9. Since CPU speeds input are peak performance, these values are derated by a factor of 4 .
10. CPUs/PGEs only communicate through the files they generate. There are no message exchanges, RPCs, or shared memory between PGEs.
11. Main memory and its effects on performance (e.g., paging) is not modeled.
12. The model simulates the intra DAAC (WAN) network as a shared resource; and the inter DAAC: (LAN) network as a switched resource. The physical configuration and protocols of the networks are not simulated; the bandwidth of the network is derated before input to account for this assumption**.**
13. The model treats secondary storage at each subsystem as network attached devices**.**
14. The model does not consider on-demand product generation**.**
15. The model does not constrain disk space.

3.7 Validation

Model validation has a multi-fold approach to insure the consistency and integrity of the models.

3.7.1 Spreadsheet/Static Model Validation

With each update of the workload baseline, the static/spreadsheet model is created from the AHWGP data, and other model input data. Then dynamic model runs are conducted for each

AHWGP instrument and for all instruments combined. The individual instrument runs are reviewed with the instrument teams to resolve anomalies, correct erroneous data/and/or modeling misunderstandings, and identify potential efficiencies. The combined instrument run, as corrected from the individual instrument runs, is then compared, component by component with the spreadsheet. The comparisons show good agreement for most components, and valid explanations for discrepancies.

3.7.2 Design/ Architecture Team Validation

The design/architecture teams for each Release/Segment review the assumptions and input for each model run, and analyze the results for credibility, feasibility and rationality. Several iterations of alternatives and/or sensitivity analysis are usual to ensure a coherent result is obtained.

3.7.3 IV&V Validation

Each version of the models and workload baseline, and the results of modeling runs are provided to the Independent Verification and Validation Contractor to perform their own analysis of the model implementation and modeling results. Findings are documented to the ESDIS Project Office.

3.7.4 Other Validation

Independent consultant(s) have provided their own model results that compare favorably with key ECS model results. Results from models developed by the DAACs will also be used. As COTS, prototype code, and algorithms become available. the model will be calibrated by benchmark measurements.

This page intentionally left blank.

4. Dynamic Model Function

The top level of the System Performance Model, Figure 4-1, is composed of three main sections, the initializing module, three load generators, and modules representing the ECS subsystems. Each module has its own set of resource arguments. These arguments have dimensionality, with each dimension corresponding to a distinct pool of resources in that subsystem at a particular site. The assignment of resources is tailorable for each dimension, and there may be multiple pools at a site.

Once a transaction is generated, it is passed to the appropriate module where actions are performed that emulate the behavior of the system design, including the allocation of resources, delays for handling the transaction, and releasing the resource upon completion. The transaction is then passed to its next destination.

4.1 Initializing Module

The initializing module of the model accepts the task request, accesses the input files, and converts them into usable tables. It then passes the task simulation on to the appropriate traffic generator. It also will act to close all files at the end of the simulation.

4.2 Traffic Generators

4.2.1 (Push) Root/External File Generator

The Root/External (Push) File Generator simulates data arrival from sources such as the AHWGP data via EOS Data and Operations System (EDOS). The file data is always passed to the Ingest Module.

4.2.2 Pull Generator

The Pull Generator simulates user demand for service. It produces transactions representing user demand for data retrieval, including subscription and ad hoc requests, user processing requests, such as subsetting, and the ingest of user data.

Transactions are analyzed in the Pull Generator and then are appropriately passed on to the Ingest Module, the User Processor, or the Data Handler.

4.2.3 V0 Migration Data Generator

This module simulates the loading of the Version 0 data as it is migrated to Version 1. As files are read, they are passed directly to the Data Handler, and thence to the Archive.

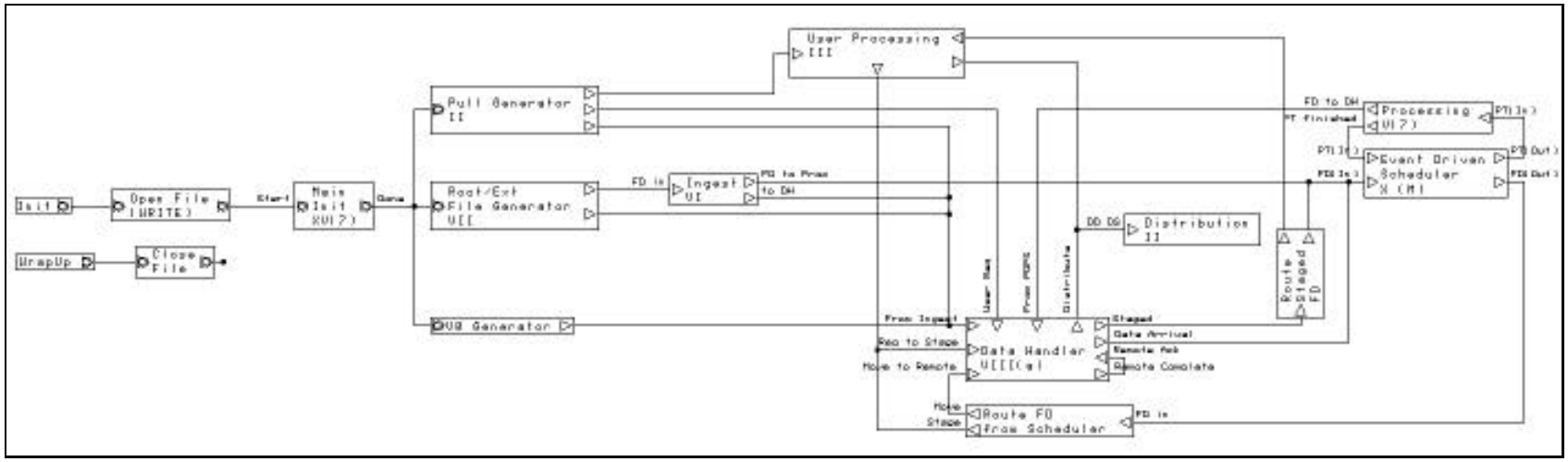

Figure 4-1. Performance Model BONeS Top Level

Notice: The diagrams in the following sections 4.3 - 4.7 are the original design and may not accurately represent the implemented design.

4.3 Ingest Module

The Ingest module emulates behavior of the Ingest subsystem. Data on physical and electronic media are accepted from external systems and users, to feed a rolling storage of Level 0 instrument data, and to transship data to other modules. File Data will be passed to either the Data Handler or the Event Driven Scheduler for further action as shown in Figure 4-2.

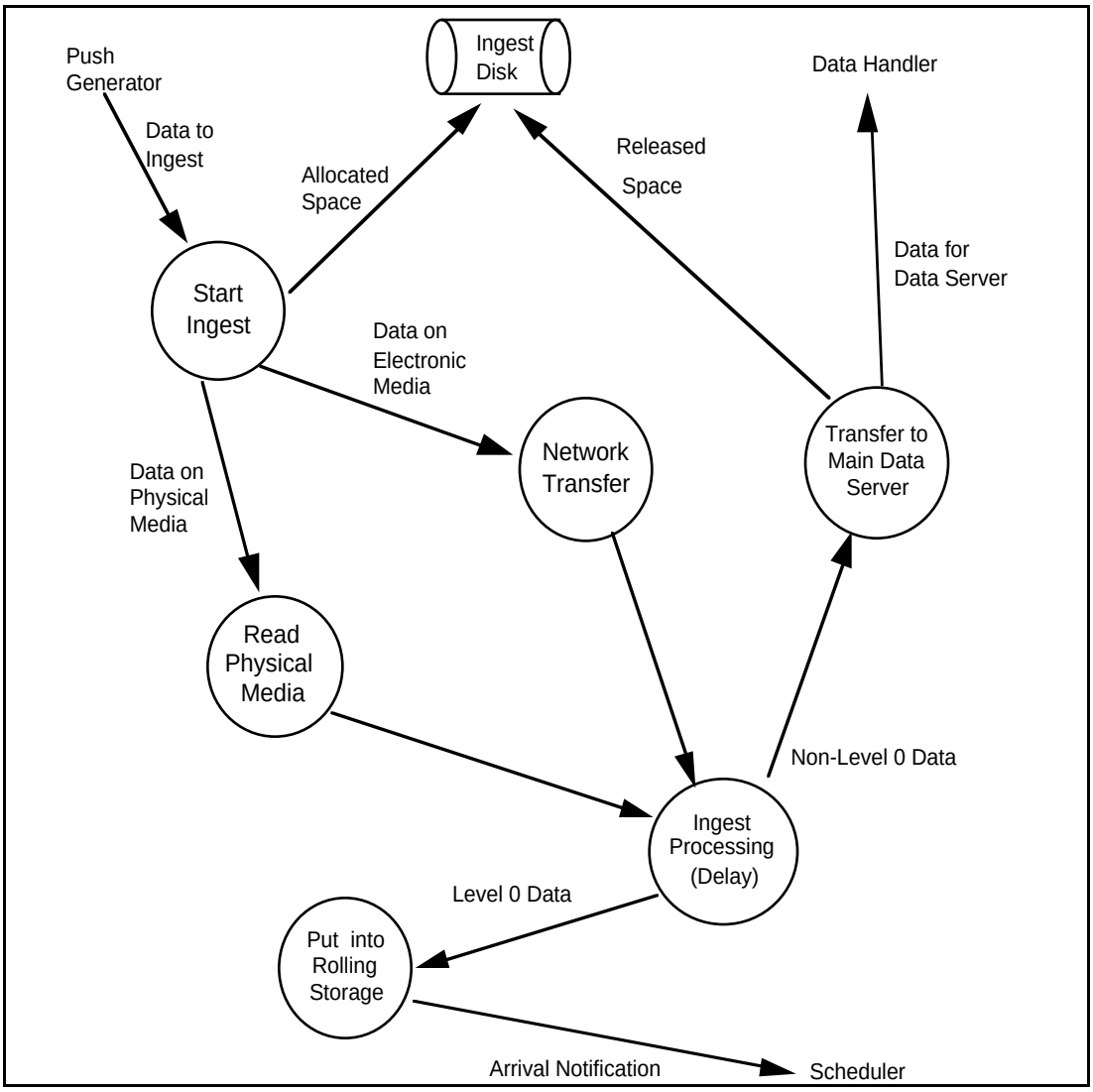

Figure 4-2. Ingest Module

4.4 Network Module

The Network Module as shown in Figure 4-3 simulates both inter DAAC (WAN) and intra-DAAC (LAN) operation.

4.5 Data Manager

The Data Manager Module encompasses the Data Handler and Distribution units.

4.5.1 Data Handler Module

The Data Handler, Figure 4-4, is the model representation responsible for storing and retrieving data from the permanent archive, for routing data to requesting modules, and for managing tiered storage resources. The module stages and destages data as requested from the pull generator, the processing unit, Ingest, and from the scheduler. Data is also staged to the Distribution Module.

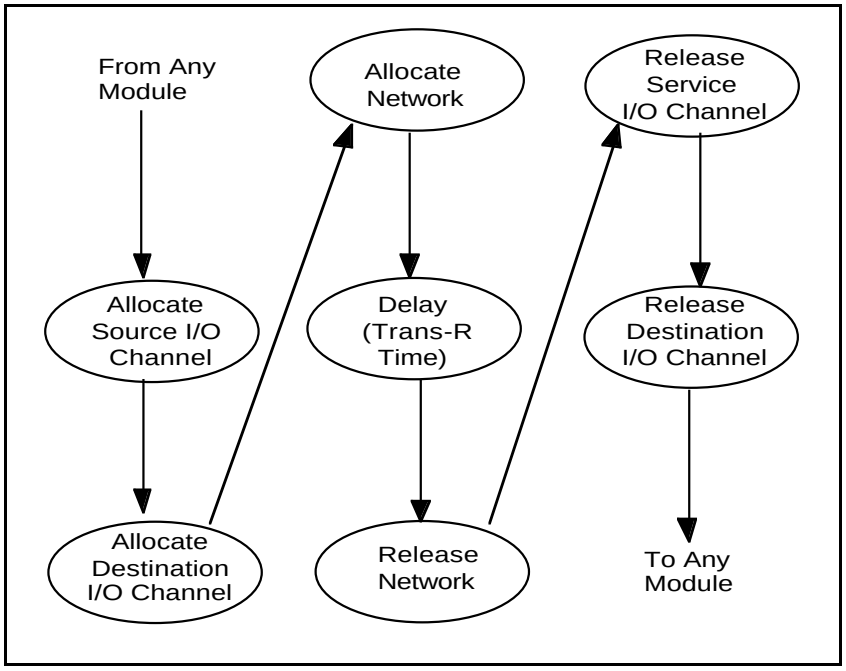

Figure 4-3. Network Module

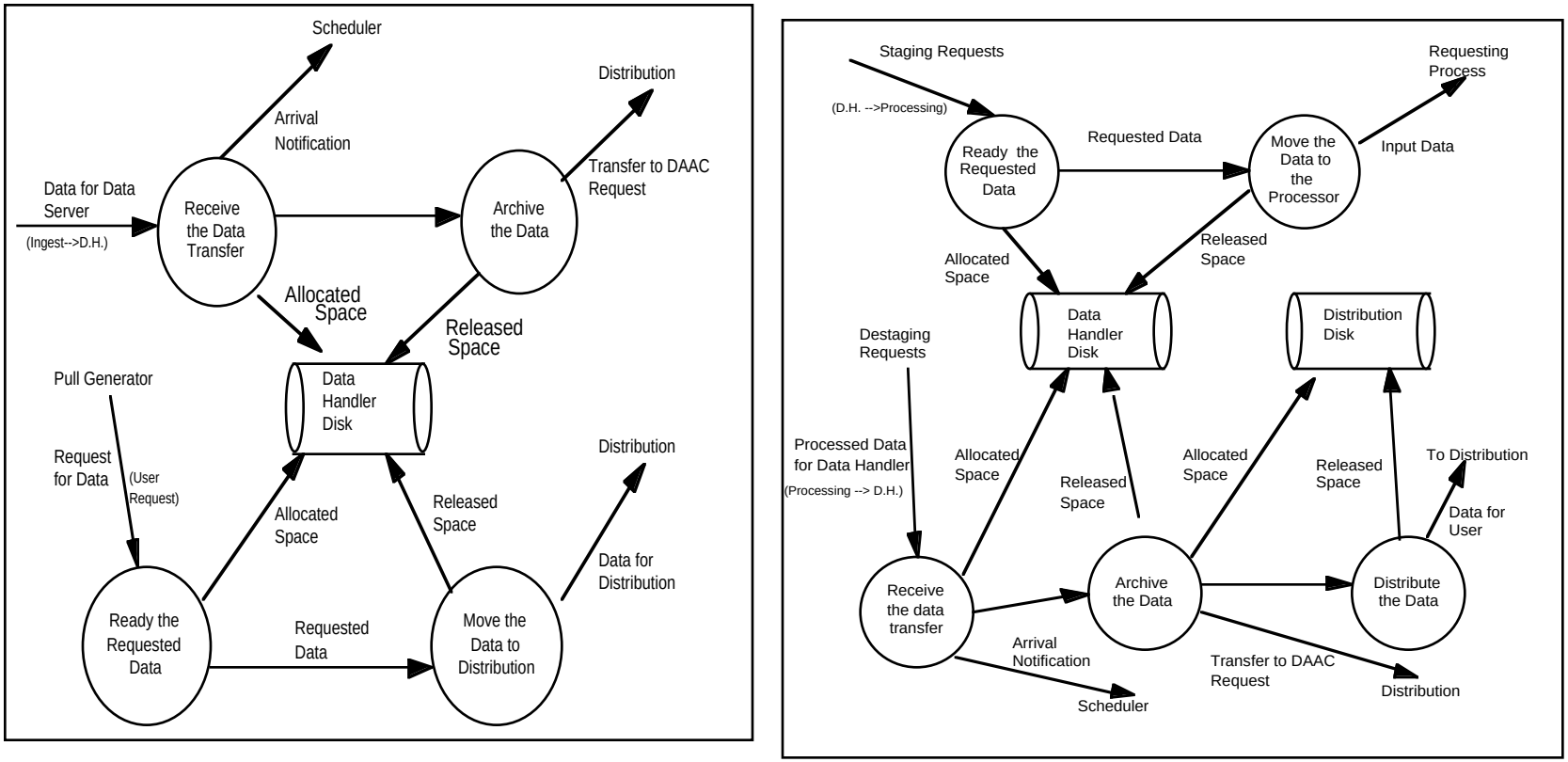

Figure 4-4. Data Handler Module

4.5.2 Distribution Module

The Distribution module, Figure 4-5, simulates network and media distribution to all users. All transactions for distribution are directed from the Data Handler. Distribution simulates the electronic transfer of data to other DAACs, to the SCFs; and, to local physical media (not shown).

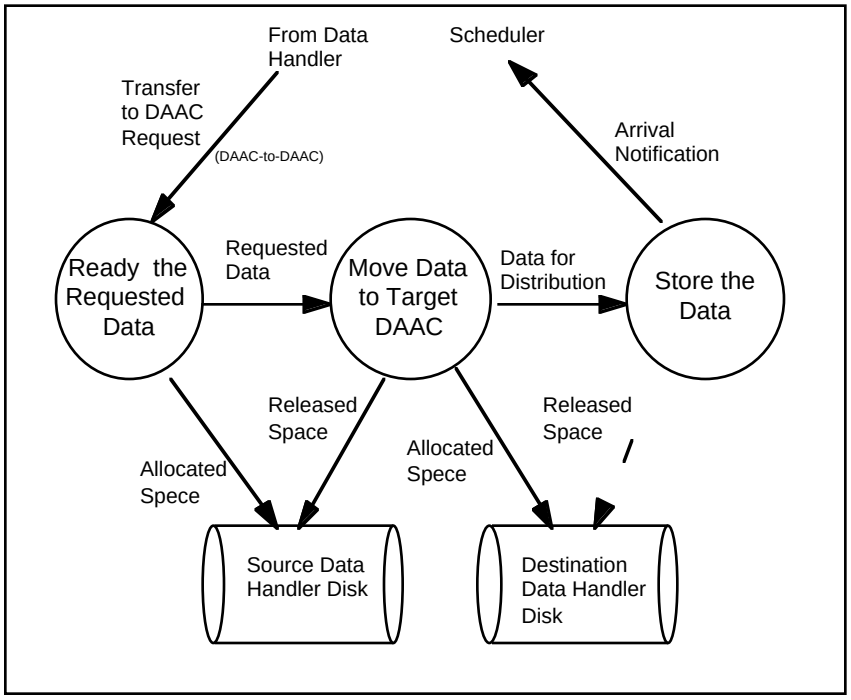

Figure 4-5. Distribution Module: DAAC to DAAC

4.6 Event Driven Scheduler Module

The Event Driven Scheduler Figure 4-6, monitors the availability of data, requests data to be staged from the Data Handler to Processing, routes newly created data to the appropriate Data Handler or Processing pool, and initiates execution of a process when all required inputs are present. The scheduler implements a simulation of the process activation rules. The primary scheduler rule is to start a process when all the input data required by the process is available on the processor staging disk and a processor is available.

4.7 Processing Module

Production of standard products is simulated by the Processing module, Figure 4-7, in conjunction with the Event Driven Scheduler and the Data Handler. Execution times and resource usage within Processing are provided by the input data (e.g., the AHWGP data).

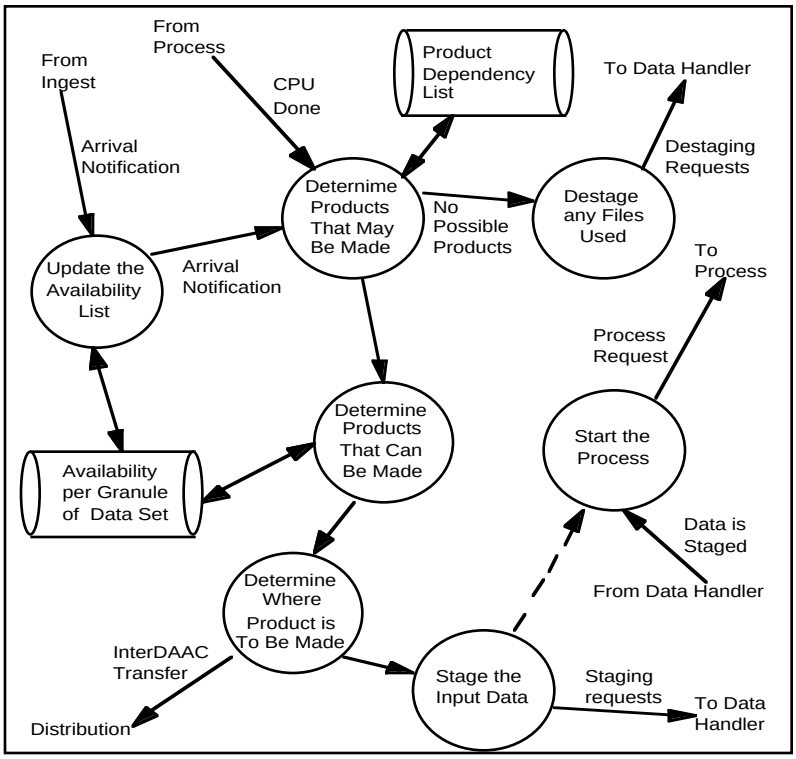

Figure 4-6. Event Driven Scheduler Module

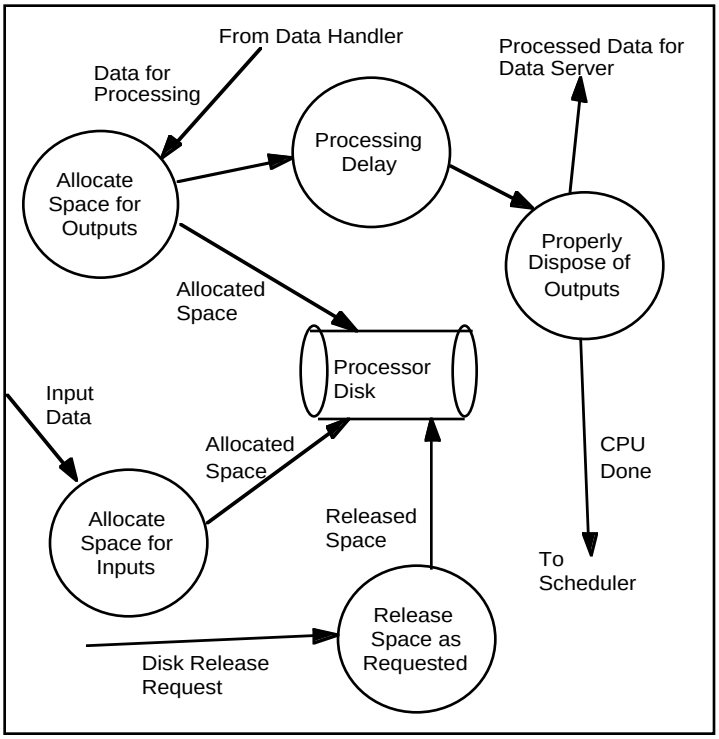

Figure 4-7. Processing Module

4.8 Failure Injection - Recovery Feature

The failure injection - recovery feature can determine the effect of hardware or software component failure on the residual resources, and the time to work off the backlog built up during the outage. At a specified time after start time, a failure may be injected into the hardware/software components of the model; at a later specified (recovery) time the hardware/software components may be replaced into the model simulation. The number of hardware/software components to be failed/recovered are specified for each time and do not have to be equal.

5. End-to-End Model

5.1 Analytic Model Overview

The end-to-end analytic model is the primary tool for determining and analyzing the response times through all the subsystems in a DAAC. The model accounts for push, pull, infrastructure loads, and distribution of products. The model provides an average or steady-state view of the ECS.

5.2 Model Input

The model input is a collection of threads partitioned into elements representing most of the work flows in the ECS and designed to account for nearly all work done in each subsystem. The thread elements include software executables and calls to other resources. Each thread and/or thread element activity has a multiplier corresponding to the frequency of invocation. and the quantity of resource used by each invocation. The input values for these activities are obtained from benchmarking, other models (e.g., the dynamic model) and from transaction estimates. The model will normally be run with the best combination of values available from any source for each activity.

5.3 Queuing Formulas

In what follows, the time unit will be *seconds* and the volume units will be *mega-* (bytes or instructions). The paradigm for a *thread* is that it has an instantiation rate (it happens so many times per second) and is composed of a list of activities. The paradigm for an *activity* is that it has three components: (1) processing, for which it has a processor from which it demands some number of million-instructions (MI); (2) network transfers, with networks Net1 and Net2 to cover the case of the transfers going over more than one network (joined by a switch), and with a number of transfers per activity, each with an average number of megabytes per transfer; and (3) a single disk transfer to a single named disk, with an average number of megabytes for the transfer. Of course, the resources have their own characteristics: number of processors and million-instructions-per-second (MIPS) rate for processing machines, megabytes-per-second data transfer rates for networks and disks, and latency times for the disks and network switches.

Arrival Statistics

To calculate the arrival rate at each resource (CPU, network link, disk), use the following. Here λ means arrival rate (in units of arrivals per second).

For CPUs:

$$\lambda_{\text{CPU}_k} = \sum_{\text{thread } i} \text{thread\_activations\_per\_second}_i \times \sum_{\substack{\text{activity } j \\ \text{in thread } i}} \begin{cases} 1, \text{ if CPU\_Processor}_j = \text{CPU}_k \\ 0, \text{ otherwise} \end{cases}.$$

For Networks:

$$\lambda_{\text{Network}_k} = \sum_{\text{thread } i} \text{thread\_activations\_per\_second}_i \times \sum_{\substack{\text{activity } j \\ \text{in thread } i}} \begin{cases} \#\text{ of network transfers}_j, \text{ if activity}_j\text{'s Net1 or Net2 is Network}_k \\ 0, \text{ otherwise} \end{cases}.$$

For Disks:

$$\lambda_{\text{Disk}_k} = \sum_{\text{thread } i} \text{thread\_activations\_per\_second}_i \times \sum_{\substack{\text{activity } j \\ \text{in thread } i}} \begin{cases} 1, \text{ if activity}_j\text{'s Disk\_name} = \text{Disk}_k \\ 0, \text{ otherwise} \end{cases}.$$

Service Statistics

To calculate average service rate μ (and average service time $1/\mu$) and the variance of the service time σ_b^2:

For CPUs (remember *MI* means Million Instructions):

$$s1_{\text{CPU}_k} = \sum_{\text{thread } i} \text{thread\_activations\_per\_second}_i \times \sum_{\substack{\text{activity } j \\ \text{in thread } i}} \begin{cases} \text{MI}_j, \text{ if CPU\_Processor}_j = \text{CPU}_k \\ 0, \text{ otherwise} \end{cases}.$$

Then $s1_{\text{CPU}_k}$ is the total number of MI that is demanded of CPU_k in an (average) second. But λ_{CPU_k} is the total number of requests on CPU_k in an (average) second. So $s1_{\text{CPU}_k} / \lambda_{\text{CPU}_k}$ is the average number of MI per request, and the average service time (in seconds) is $1/\mu = \dfrac{s1_{\text{CPU}_k} / \lambda_{\text{CPU}_k}}{\text{MIPS}_{\text{CPU}_k}}$. Next calculate

$$s2_{\text{CPU}_k} = \sum_{\text{thread } i} \text{thread\_activations\_per\_second}_i \times \sum_{\substack{\text{activity } j \\ \text{in thread } i}} \begin{cases} (\text{MI}_j)^2, \text{ if CPU\_Processor}_j = \text{CPU}_k \\ 0, \text{ otherwise} \end{cases}.$$

Then $s2_{\text{CPU}_k}$ is the total number of MI^2 that is demanded of CPU_k in an (average) second. So the

variance of the MI's requested is $Var[\text{MI}] = \dfrac{s2_{\text{CPU}_k} - \frac{1}{\lambda_{CPU_k}}\left(s1_{\text{CPU}_k}\right)^2}{\lambda_{CPU_k}}$. But since service time (seconds) $= \dfrac{\text{MI}}{\text{MIPS}_{\text{CPU}_k}}$, then we must have $Var[\text{service time}] = \sigma_b^2 = \dfrac{Var[\text{MI}]}{\left(\text{MIPS}_{\text{CPU}_k}\right)^2}$.

For Networks (here *MB* means Megabytes):

$$s1_{\text{Network}_k} = \sum_{\text{thread } i} \text{thread\_activations\_per\_second}_i \times \sum_{\substack{\text{activity } j \\ \text{in thread } i}} \text{Net\_xfers}_j \times \text{MB\_per\_xfer}_j \times \begin{cases} 1, & \text{if Net1}_j = \text{Network}_k \text{ and Net2}_j = \text{null} \\ 0, & \text{if Net1}_j \neq \text{Network}_k \text{ and Net2}_j \neq \text{Network}_k \\ 1/2, & \text{otherwise} \end{cases}.$$

Notice that if Net1 and Net2 are both present, then we count the transfers as sending half the megabytes over Net1 and half the megabytes over Net2; this keeps us from making the internetwork switch act like a store-and-forward. Then $s1_{\text{Network}_k}$ is the total number of megabytes that is demanded of Network_k in an (average) second. But $\lambda_{\text{Network}_k}$ is the total number of network transfers on Network_k in an (average) second. So $s1_{\text{Network}_k} / \lambda_{\text{Network}_k}$ is the average number of megabytes per transfer, and the average network service time (in seconds) is $1/\mu = \dfrac{s1_{\text{Network}_k} / \lambda_{\text{Network}_k}}{\text{Network\_rate}_{\text{Network}_k}}$. Now calculate

$$s2_{\text{Network}_k} = \sum_{\text{thread } i} \text{thread\_activations\_per\_second}_i \times \sum_{\substack{\text{activity } j \\ \text{in thread } i}} \text{Net\_xfers}_j \times \left[\text{MB\_per\_xfer}_j \times \begin{cases} 1, & \text{if Net1}_j = \text{Network}_k \text{ and Net2}_j = \text{null} \\ 0, & \text{if Net1}_j \neq \text{Network}_k \text{ and Net2}_j \neq \text{Network}_k \\ 1/2, & \text{otherwise} \end{cases} \right]^2.$$

Then $s2_{\text{Network}_k}$ is the total number of MB^2 that is demanded of Network_k in an (average) second.

So the variance of the MB's requested (over Network_k) is

$$Var[\text{MB}] = \frac{s2_{\text{Network}_k} - \frac{1}{\lambda_{Network_k}}\left(s1_{\text{Network}_k}\right)^2}{\lambda_{Network_k}}$$. Since service time (seconds)

$$= \frac{\text{MB}}{\text{Network\_rate}_{\text{Network}_k}}$$, then we have $$Var[\text{service time}] = \sigma_b^2 = \frac{Var[\text{MB}]}{\left(\text{Network\_rate}_{\text{Network}_k}\right)^2}.$$

For Disks (remember *MB* means Megabytes):
This analysis is very similar to the analysis for CPUs, so only the results will be given.

$$s1_{\text{Disk}_k} = \sum_{\text{thread } i} \text{thread\_activations\_per\_second}_i \times \sum_{\substack{\text{activity } j \\ \text{in thread } i}} \begin{cases} \text{Disk\_MB}_j, \text{ if Disk\_Name}_j = \text{Disk}_k \\ 0, \text{ otherwise} \end{cases}.$$

Average number of MB per disk transfer (Disk_k) = Disk_MB = $s1_{\text{Disk}_k} / \lambda_{\text{Disk}_k}$.

Average service time (disk transfer time, in seconds) = $1/\mu = \frac{s1_{\text{Disk}_k} / \lambda_{\text{Disk}_k}}{\text{Disk\_xfer\_rate}_{\text{Disk}_k}}$.

$$s2_{\text{Disk}_k} = \sum_{\text{thread } i} \text{thread\_activations\_per\_second}_i \times \sum_{\substack{\text{activity } j \\ \text{in thread } i}} \begin{cases} \left(\text{Disk\_MB}_j\right)^2, \text{ if Disk\_Name}_j = \text{Disk}_k \\ 0, \text{ otherwise} \end{cases}.$$

$$Var[\text{Disk\_MB}] = \frac{s2_{\text{Disk}_k} - \frac{1}{\lambda_{\text{Disk}_k}}\left(s1_{\text{Disk}_k}\right)^2}{\lambda_{\text{Disk}_k}}.$$

Average disk service time (seconds) $= \frac{\text{Disk\_MB}}{\text{Disk\_xfer\_rate}_{\text{Disk}_k}}$.

$$Var[\text{disk service time}] = \sigma_b^2 = \frac{Var[\text{Disk\_MB}]}{\left(\text{Disk\_xfer\_rate}_{\text{Disk}_k}\right)^2}.$$

Waiting Time Statistics

Now, why does all this matter? The answer is that the average waiting time in the queue, W_q, for *general* arrival and service distributions (G/G/1), is $W_q = \dfrac{\sigma_a^2 + \sigma_b^2 + \left(\dfrac{1}{\lambda^2}\right)(1-\rho)^2}{2\dfrac{1}{\lambda}(1-\rho)} - \dfrac{\overline{I^2}}{2\bar{I}}$, where $\rho = \lambda/\mu$ is the utilization factor, I is the distribution of idle time periods, σ_a^2 is the variance of the interarrival times, and σ_b^2 is the variance of the service times. We are assuming exponentially distributed interarrival times (M/G/1), so we can immediately write $\sigma_a^2 = 1/\lambda^2$, $\bar{I} = 1/\lambda$, and $\overline{I^2} = 2/\lambda^2$, so that $\overline{I^2}/(2\bar{I}) = 1/\lambda$. And σ_b^2 is as calculated above. This reduces to

$$W_q = \frac{\dfrac{1}{\lambda^2} + \sigma_b^2 + \left(\dfrac{1}{\lambda^2}\right)(1-\rho)^2}{2\dfrac{1}{\lambda}(1-\rho)} - \frac{1}{\lambda} = \frac{\lambda\left(\sigma_b^2 + \dfrac{1}{\mu^2}\right)}{2(1-\rho)} = \frac{\lambda E[\text{service time}^2]}{2(1-\rho)}.$$

In the case of a single queue for c processors, $c > 1$, we cannot solve explicitly for the M/G/c case, but we can for the case of exponential interarrival and service times (M/M/c). In this case, $\sigma_b^2 = \dfrac{1}{\mu^2}$, and we write $r = \lambda/\mu$ and $\rho = r/c = \lambda/(c\mu)$. Then $W_q = \dfrac{r^c \rho}{c!\lambda(1-\rho)^2} \Bigg/ \left(\displaystyle\sum_{n=0}^{c-1} \frac{r^n}{n!} + \frac{r^c}{c!(1-\rho)}\right)$. In the single server case, M/M/1, we have $c = 1$ and $r = \rho$, and both of the above formulas simplify to $W_q = \dfrac{\rho/\mu}{1-\rho}$.

Activity and Thread Time Calculations

We have now calculated the average waiting time at each resource. The total time an activity needs at a resource will then be the average waiting time plus the service time there for that activity. These service times are—(1) for processing: $\dfrac{\text{MI}_{\text{Activity}_j}}{\text{MIPS}_{\text{CPU}_k}}$; (2) for disk: $\dfrac{\text{Disk\_MB}_{\text{Activity}_j}}{\text{Disk\_xfer\_rate}_{\text{Disk}_k}} + \text{Latency\_time}_{\text{Disk}_k}$. The network service time is complicated by the

question of whether the transfer goes over two networks. Let $\text{Network\_MB}_{\text{Activity}_j}$ $= \text{Net\_xfers}_j \times \text{MB\_per\_xfer}_j$. If the transfer goes over only one network, the service time is $\frac{\text{Network\_MB}_{\text{Activity}_j}}{\text{Network\_rate}_{\text{Network}_k}}$; if there are two networks, then the service time is $\frac{(1/2)\text{Network\_MB}_{\text{Activity}_j}}{\text{Network\_rate}_{\text{Net1}}} + \frac{(1/2)\text{Network\_MB}_{\text{Activity}_j}}{\text{Network\_rate}_{\text{Net2}}} + \text{Latency\_time}_{\text{Switch}_{1,2}}$.

To calculate the average time to complete an activity, we need to add up its three components: processing time, network time, and disk time. Then the average time to complete a thread is the sum of the average completion times for its component activities.

5.4 Model Output

The output from the model is: per site/subsystem/cluster - the average number of busy processors; the average number of read/write stations, and the percentage disk utilization; the LAN utilization by site; for each thread - the end-to-end execution time, the time profile (which activities occupy how much time) and thruput (activations/day), and pull workload response time vs. arrival rate. The results are also used as a validation of the results of other models.

Appendix A. PDR--1995 Model Reference Data

A.1 PDR Model Baseline

For the Release A PDR, three epochs were given priority for simulation input data. The data in each epoch is carried into the next epochs.

1. Epoch e, January-March, 1998--for AHWGP data on TRMM(CERES & LIS).
2. Epoch g, July-September, 1998--for AHWGP data on AM 1 (ASTER. CERES. MISR, MOPITT, and MODIS)
3. Epoch k, July-September, 1999 (Epochs e and g data).

A.2 PDR Technical Baseline

The ECS system-wide Preliminary Design Review assumes a consistent set of functional and performance requirements. These requirements are established and frozen before the review in order to allow the various design teams to coordinate their designs to a consistent baseline.

The ECS has established a technical baseline to be used for the PDR design. This baseline incorporates changes to the mission baseline, as well as updates to the product baseline and user demand projections.

A.2.1 Ad Hoc Working Group (AHWGP) Data

The mission baseline consists of the following platforms (with instruments indicated in parenthesis) for the AHWGP data as of 4 January 1995:

- TRMM (CERES, LIS, VIRS, PR, TMI) - August 1997
- EOS AM-1 (ASTER, CERES, MISR, MODIS, MOPITT) - June 1998

A.2.2 User Model Data

The initial baseline for the user model was established in August, 1994 based on feedback from the SDR. The user model was upgraded, at the end of October, by including projections from the DAAC User Services Working Group, the TRMM and Landsat 7 projects, and NOAA.

This page intentionally left blank.

Appendix B. Rel A CDR, Rel B IDR--1995 Model Reference Data

B.1 CDR/IDR Model Baseline

For Release A CDR and Release B IDR, three epochs are given priority for simulation input data. The data in each epoch is carried into the next epochs, if applicable.

1. Epoch e, January-March, 1998--for AHWGP data on TRMM(CERES & LIS).
2. Epoch g, July-September, 1998--for AHWGP data on AM 1 (ASTER. CERES. MISR, MOPITT, and MODIS)
3. Epoch k, July-September, 1999 (Epochs e and g data).

B.2 CDR/IDR Technical Baseline

The ECS Release A Critical Design Review and the Release B Incremental Design Review assume a consistent set of functional and performance requirements. These requirements are established and frozen before the review in order to allow the various design teams to coordinate their designs to a consistent baseline.

The ECS has established a technical baseline to be used for the review design. This baseline incorporates changes to the mission baseline, as well as updates to the product baseline and user demand projections.

The mission baseline consists of the following platforms (with instruments indicated in parenthesis) and Launch date for the AHWGP data as of August 1995:

- TRMM (CERES, LIS, VIRS, PR, TMI) - August 1997
- EOS AM-1 (ASTER, CERES, MISR, MODIS, MOPITT) - June 1998
- Landsat 7 (ETM+, SEAWiFS II) - December 1998
- ADEOS II (SeaWinds) - February 1999
- RADARALT (MR, DFA) - CNES or GFO mission in March 1999
- ACRIMSAT (ACRIM) - June 1999
- Data Assimulation System (DAS)

ECS supports only data archive and distribution for the VIRS, PR and TMI on TRMM, and the ETM+ on Landsat 7 instrument products. ECS supports standard products processing for all other instruments. The user model data was updated with the August 1995 update.

This page intentionally left blank.

Appendix C. Rel B CDR --1996 Model Reference Data

C.1 Release B CDR Model Baseline

For Release B CDR, two epochs from the Technical Baseline are given priority for simulation input data. The data in each epoch is carried into the next epoch, if applicable.

1. Epoch k, July-September, 1999
2. Epoch m, January-March 2000 (epochs n and o are the same as Epoch m) for the Data Assimilation System (DAS) simulation only.

User Modeling/AHWGC data in the Technical Baseline is "rolled up" into the parameters used for model input. V0 migration data and instrument data from other platforms are also taken from the Baseline.

Some data, not in the Technical Baseline is obtained from the F&PRS Appendices

Data for the End-to-End Model was collected from the developers estimates, benchmarks, and the Infrastructure Design Issue Team, in addition to the Dynamic model.

All modeling through this CDR used the BONeS Version 2.6.

C.2 Release B CDR Technical Baseline

The Technical Baseline for the ECS, 210-TP-002-001, was approved in February 1996 for use in the Release B design.

This page intentionally left blank.

Abbreviations and Acronyms

AA	Advertising Agent
ACRIM	Active Cavity Radiometer Irradiance Monitor
ADEOS	Advanced Earth Observing Satellite (Japan)
ADPE	Automated Data Processing Equipment
AHWGC	Ad Hoc Working Group for Consumers
AHWGP	Ad Hoc Working Group for Products
AIRS	Atmospheric Infrared Sounder
AM-1	EOS AM Project spacecraft 1, morning spacecraft series
AMSU	Advanced Microwave Sounding Unit
ASTER	Advanced Spaceborne Thermal Emission and Reflection Radiometer (fo
BONeS	Block Oriented Network Simulation
CDR	Critical Design Review
CERES	Clouds and Earth's Radiant Energy System
CHEM	see EOS-CHEM
CNES	Centre National d'Etudes Spatiales (France)
CSMS	Communications and Systems Management Segment (ECS)
DAAC	Distributed Active Archive Center
DBMS	Database Management system
DIM	Distributed Information Manager (SDPS)
DD	Data Dictionary
ECS	EOSDIS Core System
EDOS	EOS Data and Operations System
EOS	Earth Observing System
EOS-AM	EOS Morning Crossing (Descending) Mission
EOS-CHEM	EOS Chemistry Mission
EOS-PM	EOS Afternoon Crossing (Ascending) Mission

ETM	Enhanced Thematic Mapper (Landsat 7)
GLAS	Geoscience Laser Altimeter System
HIRDLS	High-Resolution Dynamics Limb Sounder
IV&V	Independent Verification and Validation
LAN	Local Area Network
Landsat	Land Remote-Sensing Satellite
Level 0	Raw instrument data at original resolution, time ordered, with duplicate packets removed.
Level 1A	Level 0 data, which may have been reformatted or transformed reversibly, located to a coordinate system, and packaged with needed ancillary and engineering data.
LIM	Local Information Manager (SDPS)
LIS	Lightning Imaging Sensor
MB	megabyte (10^6)
MFLO	Millions of Floating Point Operations
MFLOPS	Mega (millions of)Floating-Point Operations (10^6) per second
MHS	Microwave Humidity Sounder
MIMR	Multifrequency Imaging Microwave Radiometer
MIPS	Mega (millions of) (10^6) Instructions per second
MISR	Multi-Angle Imaging SpectroRadiometer
MLS	Microwave Limb Sounder
MODIS	Moderate-Resolution Imaging Spectroradiometer
MOPITT	Measurements of Pollution in the Troposphere
NOAA	National Oceanic and Atmospheric Administration
NMC	National Meteorological Center (NOAA) network management center
PDPS	Planning and Data Processing Subsystem
PDR	Preliminary Design Review
PGE	Product Generation Executable
PR	Precipitation Radar (TRMM)
QA	quality assurance

SAGE III	Stratospheric Aerosol and Gas Experiment III
SCF	Science Computing Facility
SDPS	Science Data Processing Segment (ECS)
SeaWiFS	Sea-Viewing Wide Field-of-View Sensor
SPSO	Science Processing Support Office
SSA	Solid State Altimeter (EOS ALT);
TES	Tropospheric Emission Spectrometer
TMI	TRMM Microwave Image
TRMM	Tropical Rainfall Measuring Mission (joint US-Japan)
VIRS	Visible Infrared Scanner (TRMM)
WAN	wide area network

This page intentionally left blank.